

## 2. međuispit iz predmeta Algoritmi i strukture podataka

10. svibnja 2010.

Odgovore na prva tri pitanja napišite na svojim papirima i predajte ih u košuljici.  
Odgovore na ostala pitanja napišite na za to predviđenom mjestu uz zadatke.  
Nije dopušteno korištenje globalnih i statičkih varijabli te naredbe `goto`.

### Zadatak 1. (5 bodova)

Zadan je tip podatka **Stog** za koji su definirane funkcije za inicijalizaciju stoga, dodavanje elementa u stog te za brisanje elementa iz stoga. Prototipovi navedenih funkcija su:

```
void init_stog(Stog *stog);
int dodaj(int element, Stog *stog);
int skini(int *element, Stog *stog);
```

Funkcije `dodaj` i `skini` vraćaju 1 ako je operacija dodavanja odnosno skidanja uspjela, a 0 inače.

a) (3 boda) Napišite **rekurzivnu** funkciju **brojPojavljivanja** čiji je prototip:

```
int brojPojavljivanja (int element, Stog *stog);
```

Funkcija treba vratiti broj pojavljivanja zadanog elementa na stogu.

b) (2 boda) Napišite funkciju **ostaviVisestruke** čiji je prototip:

```
void ostaviVisestruke(Stog *stog, int brPojavljivanja);
```

koja će korištenjem funkcije **brojPojavljivanja** na zadanom stogu ostaviti samo one elemente koji se pojavljuju **više ili jednako** od `brPojavljivanja`.

Primjer: nakon poziva funkcije **ostaviVisestruke (stog, 3)** stog s početnim vrijednostima 3 3 5 6 3 5 3 2 5 4 2 8 5 sadržavat će 3 3 5 3 5 3 5 5 jer se ti elementi pojavljuju tri ili više puta.

### Zadatak 2. (5 bodova)

Zadan je tip podatka **Red** za koji su definirane funkcije za inicijalizaciju reda, dodavanje elementa u red te za brisanje elementa iz reda. Prototipovi navedenih funkcija su:

```
void init_red(Red *red);
int dodaj(int element, Red *red);
int skini(int *element, Red *red);
```

Funkcije `dodaj` i `skini` vraćaju 1 ako je operacija dodavanja odnosno skidanja uspjela, a 0 inače.

Napišite funkciju čiji je prototip:

```
void izbaciDuple(Red *red);
```

koja će iz zadanog **sortiranog** reda ukloniti sva dodatna pojavljivanja pojedinog elementa.

Primjer: Red brojeva 3 3 3 5 6 6 7 7 8 9 9 9 funkcija će promijeniti u 3 5 6 7 8 9.

### Zadatak 3. (4 boda)

Zadan je niz brojeva: **1, 5, 7, 4, 3, 6, 8, 2, 9, 0**

Ilustrirajte uzlazno sortiranje zadanog niza brojeva (ispišite niz nakon svake zamjene dvaju elemenata):

- (2 boda) Algoritmom **shell sort** za slijed koraka {4, 2, 1}. Kako izgledaju 4-, 2- i 1-sortirani nizovi?
- (2 boda) Algoritmom **quicksort**. Stožer odaberite metodom aproksimacije medijana temeljem početnog, krajnjeg i središnjeg člana polja.

### Zadatak 4. (2 boda)

Ispišite vrijednosti (od izlaza prema ulazu reda) koje se nalaze u redu nakon što se obavi funkcija **func**. Funkcija **func** kao parametar prima red s vrijednostima: 1, 2, 3, 4, 5, 6, 7, 8. Funkcija **dodajURed** dodaje vrijednost argumenta *el* u red.

```
void func (Red *red){
    int el;
    if (skiniIzReda(&el, red)) func(red);
    if (el % 2) dodajURed(el, red);
}
```

---

### Zadatak 5. (2 boda)

Koristeći prostor nad crtama, napišite naredbe **silaznog Insertion sorta** koje nedostaju.

```
void InsertionSort (tip A [], int N) {
    int i, j;
    tip pom;
    for (i = 1; i < N; i++) {
        pom = A[i];
        _____
        _____
        A[j] = pom;
    }
}
```

### Zadatak 6. (2 boda)

Stog je definiran na sljedeći način:

```
typedef struct {
    int vrh, polje[MAXSTOG];
} Stog;
```

Koristeći prostor nad crtama, napišite naredbe koje nedostaju za ispravan rad funkcije koja dodaje element na stog.

```
int dodaj(int stavka, Stog *stog){
    if (stog->vrh >= MAXSTOG-1) return 0;
    _____
    _____
    return 1;
}
```

# Rješenja

## 1. zadatak

```
int brojPojavljivanja(int element, Stog * stog){
    tip pomelem;
    int broj=0;
    if (skini(&pomelem, stog)){
        if (pomelem==element) broj = 1;
        broj=broj + brojPojavljivanja(element, stog);
    }
    else
    {
        return 0;
    }

    dodaj(pomelem, stog);
    return broj;
}

void ostaviVisestruke(Stog * stog, int brPojavljivanja){
    int n;
    Stog pom1, pom2;
    init_stog(&pom1);
    init_stog(&pom2);

    while(skini(&n, stog)){
        dodaj(n, &pom1);
        dodaj(n, &pom2);
    }

    while(skini(&n, &pom1)){
        if (brojPojavljivanja(n, &pom2)>=brPojavljivanja) dodaj(n, stog);
    }
}
```

Rješenje s jednim pomoćnim stogom:

```
void ostaviVisestruke(Stog * stog, int brPojavljivanja){
    int n;
    Stog pom1;
    init_stog(&pom1);

    while(skini(&n, stog)){
        if (brojPojavljivanja(n, &pom1) + brojPojavljivanja(n, stog) +1) >=brPojavljivanja)
            dodaj(n, &pom1);
    }

    while(skini(&n, &pom1)){
        dodaj(n, stog);
    }
}
```

## 2. zadatak

```
void IzbaciDuple(Red *red){

    Red pom; int element, temp;

    init_red(&pom);

    skini(&temp, red);
    dodaj(temp, &pom);
    while(skini(&element, red)){
        if (temp != element) {
            temp = element;
            dodaj(element, &pom);
        }
    }
}
```

```

    }
}
while(skini(&element, &pom)){
    dodaj(element, red);
}
}

```

### 3. zadatak

Rješenje:

a)
1 5 7 4 3 6 8 2 9 0
1 5 7 2 3 6 8 4 9 0
1 5 7 2 3 0 8 4 9 6
1 0 7 2 3 5 8 4 9 6
1 0 7 2 3 5 8 4 9 6
1 0 3 2 7 5 8 4 9 6
1 0 3 2 7 4 8 5 9 6
0 1 3 2 7 4 8 5 9 6
0 1 2 3 7 4 8 5 9 6
0 1 2 3 4 7 8 5 9 6
0 1 2 3 4 7 5 8 9 6
0 1 2 3 4 5 7 8 9 6
0 1 2 3 4 5 7 8 6 9
0 1 2 3 4 5 7 6 8 9
0 1 2 3 4 5 6 7 8 9

b)
1 5 7 4 3 6 8 2 9 0
0 5 7 4 1 6 8 2 9 3
0 5 7 4 9 6 8 2 1 3
j i
0 1 7 4 9 6 8 2 5 3
3 4 9 6 8 2 5 7
3 4 9 5 8 2 6 7
i j
3 4 2 5 8 9 6 7
j i
3 4 2 5 6 9 8 7
3 2 4 5
j i
7 8 9

### 4. zadatak

Rješenje: 7, 5, 3, 1

### 5. zadatak

```

for (j = i; j <= 1 && A[j-1] > pom; j--)
    A[j] = A[j-1];

```

### 6. zadatak

```

stog->vrh++;
stog->polje[stog->vrh] = stavka;

```